|  |
| --- |
| Accenture |
| Create wbs user guide |
|  |

# Introduction

This document aims at providing a simple user guide for the create\_wbs script. The document is internal to Accenture Mobility Services. It is targeted at software developers maintaining the script and project managers managing project plans in RTC.

# Scope

In RTC, work items can be linked through a parent child relationship, allowing for pieces of work to be broken down in smaller refined work items. This is referred to as the work breakdown structure (WBS). Unfortunately the exported file does not visually capture this relationship; thus making it difficult to navigate the work items. The create\_wbs script enhances exported data by sorting the work items and grouping them in a work break down structure.

# Prerequisites

Create\_wbs is written in python and json and is available from the rtc\_create\_wbs git repository. Below are the software prerequisites for using this script:

1. Git installed
2. Access to the AMS HOF git infrastructure
3. RTC exported csv file with the following mandatory fields:
   1. Planned for
   2. Priority
   3. Rank (relative to Priority)
   4. Id
   5. Parent
   6. Type
   7. Story points

## Python

1. Download Python 3.3.5 from <https://www.python.org/download/releases/3.3.5/>
2. Follow installation process and remember the file path
3. Open Control Panel/System and Security/System/Advanced system settings
4. On the Advanced tab, click on Environment Variables
5. Click on System variables
6. Append your chosen path following Path (e.g ";c:\Python30" if you have installed on your c drive)

## Xlswriter

1. Download xlswriter from <http://xlsxwriter.readthedocs.org/getting_started.html#getting-started>
2. At command prompt, run command python setup.py install

## Git Repo

1. At command prompt, navigate to a folder where the scripts will be stored.
2. Pull down the scripts using the git command

git clone https://*[your user id]*@ams.accenture.com/gerrit/p/rtc\_wbs.git

1. Run the command python create\_wbs.py *[name of input csv file] [name of output file]*

# Voila

## Creating a Query in RTC

## Running RTC query/export to xlsx

## The output file

Create\_wbs.py generates an xlsx file with 3 tabs:

1. Input tab is populated with the raw data from the input file
2. Ranked tab is populated with the work items sorted on fields “Planned For”, “Priority”, “Rank” and “ID”
3. Grouped tab displayed the work breakdown structure with the child work items grouped under the parent work item.

## The config file

This script allows the user to configure the formatting and ranking order that is required for determining the grouping. This can be easily done through a json file called config.json.

The config file consists of an array of dictionaries where each dictionary corresponds to a piece of configurable data.

### Plans

The entry "Planned For" is a dictionary consisting of the names of the project plans and their order or priority. The script sorts the plans in ascending order. The name of the plans must correspond to the names in the RTC export and assigned with the correct priority:

"Planned For":{

[“name of plan”], [priority number]

},

### Priority

The “Priority” entry is a dictionary consisting of priority and the enumerated values. The script ranks the work items in ascending order of the enumerated values, where 1 is ranked highest in the list.

"Priority":{

"High":1,

"Medium":2,

"Low":3,

"Unassigned":4

}

It is not recommended to change this section unless the priority data changes in the RTC export.

### Format

The “Format” entry is a dictionary consisting of configuration for the rows which require distinct text formatting in the output file.

entry :[Font name ,size ,bold, italic, colour]

The font type, size, bold, italic and colour of the text can be configured for the following rows capability, defect, epic, feature, story and header.

"Format":{

"Capability":["Times New Roman",10,"False","False","black"],

"Defect":["Times New Roman",10,"False","False","black"],

"Epic":["Times New Roman",14,"False","True","black"],

"Feature":["Times New Roman",14,"False","False","black"],

"Story":["Times New Roman",12,"False","False", "black"],

"Header":["Times New Roman",15,"True","False","blue"]

}